לינק לGitHub של הפרויקט: https://github.com/evghega/Collind

*ניתן למצוא שם גם את הUnitTest.

הדרישות שבותחו :

22) **כמטופל אני יכול להתחיל משחק כדי להיבחן** – *(22.1)*מומש כפתור התחל משחק בGUI שלאחר לחיצה עליו *(22.2)*עוברים לעמוד המשחק, *(22.3)*מומשה פונקציה אשר יוצרת בDB עמודה חדשה למשחק.

23) **כמטופל אני יכול לזרוק את הקובייה כדי לדעת איזה צורה לחפש** –*(23.1)* מומש כפתור לזריקת קובייה, *(23.2)* מומשה פונקציה רנדומלית המגרילה צורה ומציגה את הצורה למשתמש.

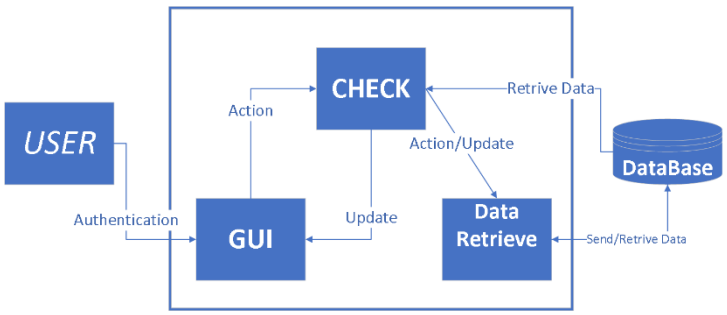
25) **כמטופל אני יכול לפתוח קלף כדי להתקדם במשחק** – *(25.1)*מומש כפתור לפתיחת קלף מומשה *(25.2)*פונקציה רנדומלית המגרילה קלף ומציגה אותו למשתמש.

24) **כמטופל אני צריך להגיד כמה מהצורה שיצאה בקובייה אני רואה בקלף** –*(24.1)* מומש GUI של הצגת קלף, *(24.2)*GUI לבחירת תשובה , *(24.3)*מומשה פונקציה הבודקת האם התשובה שנבחרה נכונה ומעדכנת את הDB.

תשובות לשאלון אינטגרציה :

a) מהנדסי תוכנה יבצעו את עבודתם בעקביות לפי צרכי הציבור וטובתו.
יוודאו שהמוצרים וההתאמות שהם עורכים להם, יעשו עפ"י התקנים המקצועיים הגבוהים ביותר האפשריים.
יקדמו את היושרה והמוניטין ישמרו על סודיות במידת הצורך , ישתמשו רק בדברים מאושרים ע"י הלקוח.
על עובדי המערכת לשמור על פרטיות המשתמשים/ לקוחות , לא לערב גורמים חיצוניים שאינם קשורים לפיתוח המערכת או ללקוחות .
על העובדים לשמור על אמינות וערכיות לא להשתמש בפיתוחים/פטנטים של אחרים ולא להשתמש בכוחם להפצה או גניבה של מסמכים אישיים.
על עובדי המערכת לשמור על חלוקה הוגנת בין חברי הפרויקט, לתמוך אחד בשני ולפרגן.
נשמור על פרטיות המאובחנים ולא נשתמש בדוחות שאינם מאושרים לשימוש ע"י מאבחני אנשים עם לקויות ראייה .
נשמור על יושרה ועצמאות, פיתחנו את המשחק בעזרת רעיונות של חברי הצוות, לא נשתמש בפטנטים של אחרים אלא שלנו בלבד.
בפרויקט זה שמרנו על חלוקה שווה בין חברי הצוות, תוך כדי תמיכה ועזרה אחד לשני בעת הצורך .

b) הקשר בין ארכיטקטורה לדרישות לא פונקציאליות:
במערכת שלנו השתמשנו בסביבת העבודה pycharm עם ממשק pygame אשר בעזרתו פיתחנו את התצוגה שלנו Gui.
את הDB שלנו מימשנו בעזרת Excel שם ישמרו כל נתוני המשתמשים בין אם זה התחברות לתוצאות המשחקים.
האבטחה של המערכת תהיה השימוש בשם המשתמש וסיסמא לצורכי התחברות



c) כולנו עובדים על אותו פרויקט, באותו מקום ובאותו זמן.
רוב העבודה נעשתה בזמן שיושבים יחד במקום העבודה ועושים את העבודה בצורה נוחה יותר.
חלקנו את העבודה חצי מהזמן שניים עבדו על כתיבת הקוד ושניים על חיפוש מידע, ובשאר הזמן החלפנו את התפקידים.
אחד עבד על חיפוש מידע שקשור לבסיס הנתונים, אחד על טכנולוגיות חדשות אחרות.
כל הזמן שמרנו על שיתוף המידע אחד של השני והתייעצנו על איך כדאי באמת לפעול ולעבוד.
אלעד – התחיל בבניית GUI המערכת בpyGame.
צליל – כתבה פונקציות וקישרה אותן לבסיס הנתונים.
שניר – ניווט את החלוקה, יצר תרשימים, הוסיף פונקציות.
יבגני – כתב קוד וחיפש מקורות.
חילקנו את שאלון האינטגרציה בין חברי הצוות כל אחד ענה על שאלה אחת מהשאלון ונעזרנו אחד בשני.
השתמשנו בכל אמצעי אלקטרוני שעמדו לרשותנו , מחשבים, פלאפונים , עכברים (בכדי לשמור נוחיות).
השתמשנו גם בכתיבה במחברות בכדי לציין נקודות חשובות.
במידה ומישהו נתקל בבעיה חברי הצוות ישר קמו לעזרתו.
לאחר כל ביצוע משימה היינו שוב עוברים על החלוקה וביצוע המשימות אם זה להמשיך בתחום התכנות או חיפוש מקורות.
3 מקרים בהם העבודה ב mob programming שיפרה את תהליך העבודה :

1) כשחברי הצוות יושבים יחד פיזית ועובדים יחד בו זמנית משפר את העבודה מפני שאפשר להיעזר אחד בשני במידת הצורך .
2) התהליך עזר בכך שהשתמשנו יחד בכמה מכשירים שונים בכדי לבצע משימות שונות בו זמנית וקידום תהליך העבודה.
3) חלוקת המשימות תרמה לניהול הפרויקט בצורה מסודרת ויעילה.
כל אחד עשה את החלק שלו, שמרנו על הסדר ובכך זה תרם לעמידה בזמנים.

d) חלק מהיתרונות של בדיקת תוכנה ובעיקרם בדיקות יחידה שהם לרוב הרמה הראשונה של בדיקת יחידה כאשר [1] בודקים שהפונקציה (1) עושה כל מה שמתאר המפרט שלה, (2)לא עושה דבר שהיא לא צריכה לעשות. המוטיבציה לבדיקת יחידה היא העובדה שהעלות למצוא ולתקן בעיה במהלך בדיקת היחידה הוא הרבה יותר זול מאשר למצוא ולתקן בעיות שנמצאו במהלך בדיקות האינטגרציה, בדיקת מערכות או בתהליך היצור. יתר על כן, בדיקות יחידה מקלות על בדיקות הרגרסיה. מכיוון של פעם שתוכנה משתנה, ניתן לעשות בדיקה שמשהו קודם לא נהרס. החסרונות העיקריים בבדיקות התוכנה הם [2] כאשר לא מבצעים בדיקות יחידה איכותיות זה יוביל לאיכות תוכנה נמוכה שתערער את הצלחת המוצר. בדיקת תוכנה לקויה תוביל לאובדן האמון בתוכנה ולריקבון של התהליכים . לעיתים קרובות הצוות ינסה "לחסוך בזמן" על ידי ביצוע מועט של בדיקות יחידה ואף לא בכלל. והסיכון לבעיות תהיה גדולה יותר . במידה ויש בדיקה מרובה מידי , הדבר יכול להוביל לבדיקה מרובה ולא הכרחית אשר תוביל להאטה בקצב הפיתוח.

---

[1] https://ieeexplore-ieee-org.ezproxy.sce.ac.il/stamp/stamp.jsp?tp=&arnumber=7051868 עמוד 29, נושא 3.
[2] https://resources.sei.cmu.edu/asset_files/WhitePaper/2012_019_001_495381.pdf עמוד 76